CKME136 – Capstone Project

Comparative Analysis for Walmart Sales Forecast

Adrian Maurice Chun Wang Wong (501030070)

27/07/2020

Github Link: https://github.com/adrianmwong/CKME136-Capstone-Project

Ryerson University

Table of Contents

Comparative Analysis for Walmart Sales Forecast

Introduction

Walmart Inc. is a multinational retail company with more than ten-thousands stores worldwide—selling all sorts of products including, grocery, household items, furniture, clothing, jewelry, electronics, and more. It is always hard to predict accurate sales for a huge company like Walmart, as many hidden factors could affect sales. Since the past centuries, various predictive methods and models have developed, and Walmart Inc. must choose an accurate method for predicting future sales as it influences management decisions. With an accurate prediction of sales, management could better allocate resources for marketing and finding out which departments are doing poorly to maximize profit. Using Walmart Dataset from Kaggle as a case study, this project seeks to address the following question:

With the use of regression models, does the prediction of sales become more accurate than traditional time series methods?

By computing time series methods: Auto Regressor (AR), Moving Average (MA), Auto-Regressive Integrated Moving Average (ARIMA), Seasonal ARIMA (SARIMA), and Holt-Winters' Method (HW). Then, regression methods, including Decision Tree, Random Forest, and XGBoost. All of the mentioned above methods would be compare against Mean Absolute Error (MAE), Mean Square Error (MSE), Root Mean Square Error (RMSE). For the calculation of algorithms, Python in Jupyter Notebook would be used. The best model would be used to predict future sales.

Literature Review

Businesses have been trying to utilizing math and different technique to predict sales, with accurate prediction, it is beneficial for them to prepare how much to order and for marketing. Throughout the centuries, many methods are being developed and used for predictions. It is always a hard choice to choose what methods as each technique has its strength and limitation (Mentzer & Moon, 2004; Lasek et al., 2016).

Conventional time series methods such as ARIMA, SARIMA and HW are always used for sales forecasting. It is not surprising that a researcher found out that more than 50% of the research papers on predictive analytics for the supply chain used the time series forecasting approach (Bonnes, 2014). Time-series method analyzes time series sequences and investigates what the parameter statistics and other characteristics are, and used that to predict future values using observed historical value.

On the other hand, some researchers argue that conventional regression models such as decision tree, random forest, and XGBoost should be used for sales prediction (Wu et al.,2018; Hülsmann et al., 2012). Rather than just based on previous year data to predict, they believe that sales prediction is instead a regression problem. It should be more like testing theories, checking whether the current value in one or more independent time series influences the current value in another time series due to multiple attributes (Pavlyshenko, 2019).

Many works of literature have written in the topic related to sales prediction, below are of some research paper on both time series and regressions models, the following literature review would be used to understands different techniques, their strengths and weaknesses.

Comparison of AR, MA, ARIMA, SARIMA, and HW (Time Series):

Prasetyo et al. (2019) compared the three models of AR, MA, and ARIMA for shoe sales prediction in Indonesia. Using the MAE as the performance metric, the authors concluded that ARIMA performed the best.

Sumer et al. (2019) compared ARIMA and SARIMA with a Turkish electric company to predict demands. They used RMSE, MAE, and Mean Absolute Percentage Error (MAPE) as performance metrics. Concluding that SARIMA performs better as it considers the seasonality effect.

Udom (2014) was predicting sales of five products from a plastic industry in Thailand using MA, SARIMA, and HW. Using MAPE as a performance metric, the author concluded that SARIMA was the best.

Purthan et al. (2014) compared the actual prediction of Indian motorcycle sales with SARIMA and HW. Using MSE, MAE, and MAPE as performance metrics, the results show that HW provides better accuracy and precision.

Makatjane & Moroke (2016) were predicting sales for monthly car sales in South Africa using 19 years of data with HW and SARIMA. MAE, MAPE, and MSE were used as a performance metric, and they concluded that HW methods outperform SARIMA.

Comparison of Decision Tree Regressor, Random Forest Regressor, and XGBoost (Regression):

Jain et al. (2015) used linear regression, random forest regression and XGBoost to predict a German drug store retail sale. The evaluation metric was Root Mean Square Percentage Error (RMPSE), and the results show that XGBoost was the best.

Knott et al. (2015) also used the German drug store for model comparison. In addition to the model that is listed in Jain et al. (2015), the researchers even Hidden Markov Model and Recurrent Neural Nets to predict the sales. Using RMPSE as the evaluation metric, the results also show that XGBoost performs the best.

Massaro et al. (2018) compared the same dataset as in this research paper, which is Walmart sales. Their focuses are on regression and deep learning models including Neural Network, Support Vector Machine (SVM), k-NN, Decision Tree, Random Forest, and XGBoost by comparing with performance metrics of average absolute errors, relative average errors and correlations. Their finding suggested that Neural Network performed the best as it has the highest correlation, lowest average absolute error, and smallest relative average error. Second is XGBoost, then Random Forest, then Decision Tree and the rest of regression models.

Catal et al. (2019) also used Walmart dataset to compare the different models. In their research paper, they used time series methods such as ARIMA, SARIMA, Seasonal ETS, regression models such as Bayesian, Linear, Random Forest, XGBoost, and Neural Network Regressor. RMSE and MAE were the performance evaluator. Interestingly, the result shows that Random Forest Regressor outperforms all the other methods, and SARIMA was ranked second, then XGBoost. Contrast to Massaro et al. (2018); Neural Network Regression performed the worst among the models.

Comparison of Time Series and Regression Approaches:

Regardless of using the time-series approach or regression approach, one thing for sure is that there are never the “best” methods, as each dataset is different, and most of these methods and techniques have their strengths and weaknesses. Testing and comparing the results are needed to see, which yields the best results. Therefore, in this research paper, the main focuses are to determine which methods work the best with the Walmart Sales forecast and would be ranked based on MAE, MSE, and RMSE.

Dataset

In the Kaggle dataset[1], Walmart has provided historical weekly sales data extracted from the 45 Walmart store from 05/02/2010 to 26/10/2012, a total of 421,570 instances. Each of the stores contains around 90 departments. Additional to date and weekly sales amount, the dataset also included other variables such as store size, type, temperature, fuel price, consumer price index (CPI), unemployment date, and Walmart promotional markdown event date. The challenge for this paper is to predict department-wide sales for each store and how promotional markdown events (Super Bowl, Labor Day, Thanksgiving, and Christmas) affect the prediction, as Walmart decided that the holiday weeks are weighted five times higher than non-holiday weeks.

There are five CSV files included in the dataset: Features, Stores, Train, Test, and SampleSubmission (will not be used, as it is for Kaggle submission only). Total of 16 attributes noted, and description of the characteristics listed below (Table 1):

Attributes:	Description:
Store	This is the store number, and it ranged from 1-45
Dept	It is the department number, it ranged from 1-99, for different categories of items.
Date	The Week
IsHoliday	Whether that specific week has a special holiday in it
Weekly_Sales	Store weekly total amount in USD
Temperature	The average weekly temperature of the particular store region in Fahrenheit
Fuel_Price	Cost of Fuel of the particular store region in USD
MarkDown1-5	Anonymized Data related to Walmart's promotional markdown is only available after November 2011, and not all stores have it
CPI	Consumer Price Index of specific store region for the week
Unemployment	The unemployment rate of particular store region for the month
Type	Type of the store, A, B or C
Size	Size of the specific store measured in square feet

Table 1: Attributes and Descriptions

[1] https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting

Methodology

In this section, a general description of the methodology for this research paper would be discussed and followed by the data analytics techniques that would be use and performance evaluation metrics that would apply to these data analytics techniques.

This research paper aims to use historical data to predict the future sales of Walmart stores and departments using traditional time series approach, regression approach, and neural network approach. The methodology approach is illustrated in figure 2.

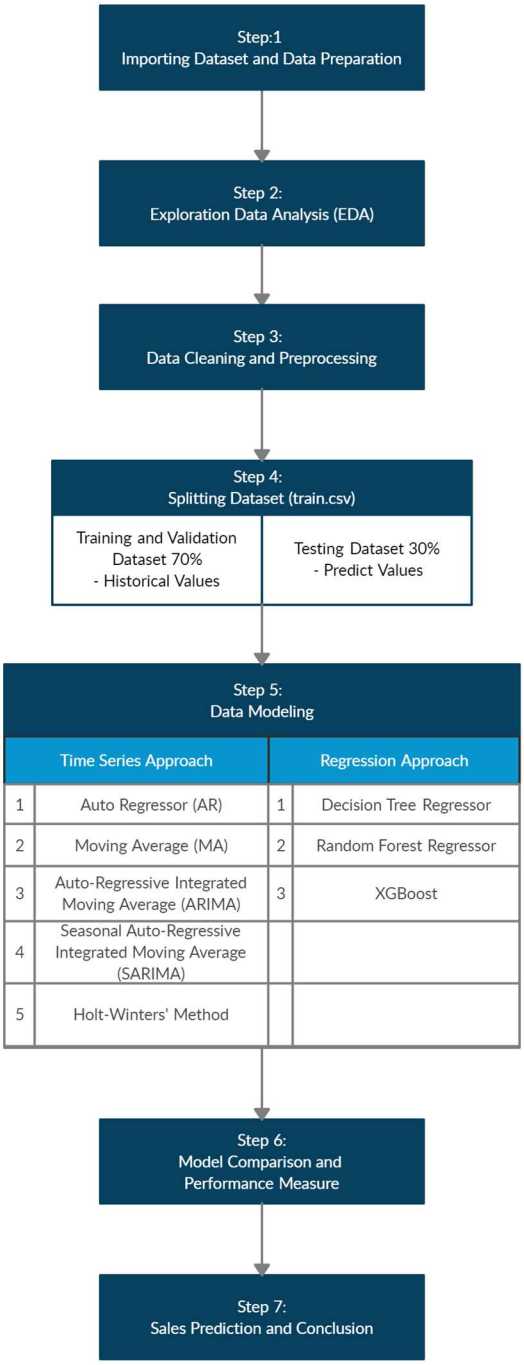

Fig.2: Schematic representation of the methodology

Step 1: Data import and data preparation. The CSV files would be loaded the dataset into Jupyter notebook. Then it would be reviewed for understanding purposes only, such as how many variables are in it and the number of instances. The relevant CSV files would be merged with the inner join.

Step 2: Exploratory Data Analysis (EDA). After merging of the CSV file, EDA would be performed. Descriptive statistics would be used to investigate the count, mean, minimum, maximum, interquartile ranges, and number of NAs. The paper would also examine the relations between sales (dependent variable) and other independent variables with visual aids.

Step 3: Data cleaning and preprocessing. Based on the previous analysis of dependent variables and independent variables. The data would be clean, including replacing NAs, Boolean and category type data to integers, and set Date as Index. Then, a correlation heat map would be used to see the correlations of all the attributes. In the end, low correlations attributes would be removed from the dataset.

Step 4: Splitting dataset. The CSV file included all historical data split into two parts in a ratio of 70:30 by date for the training set and testing set, respectively.

Step 5: Data modeling. The training set would fit all the suggested models in time series approaches, regression approaches, and neural network approaches. In the time-series approach, conventional methods included AR, MA, ARIMA, SARIMA, and HW. In the Regression approach, traditional methods included Decision Tree Regressor, Random Forest Regressor, and XGBoost would be used.

Step 6: Model comparison and performance measure. After each of the methods developed, all models would be used to compare against each other with performance metrics, such as MAE, MSE, and RMSE.

Step 7: Sales Prediction and conclusion. Based on the previous level, the lowest MAE, MSE, and RMSE score method would be used to predict future weekly sales. Recommendation and limitation would be used to conclude the research paper.

Import Dataset and Data Preparation

Since the Dataset section above already investigated the nature of the attributes, here would just show where they located in each CSV file. Below are the four CSV files, including their attributes and number of instances. The train.csv file is the main CSV file that would be used for data modeling and testing. The feature.csv and store.csv file are additional features that Walmart included that may affect the weekly sales data. Therefore, it would be merged to train CSV files later on. The test.csv is the actual file used to predict sales, a total of 39 weekly sales data would be predicted in a later section.

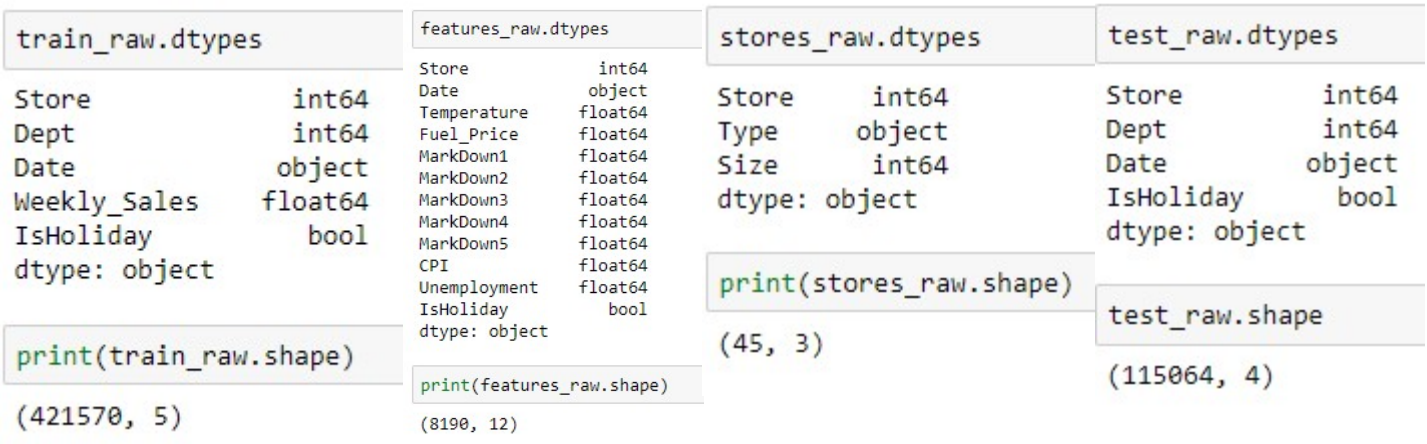

Fig.3: Shapes of the four CSV files

The method used for merging train.csv, feature.csv and stores.csv is inner join based on the attribute of "Store", "Date", and "IsHoliday". Below are the first five instances for preview (Table 4). Total of 421,570 instances 16 attributes.

```
train_merged.head(5)
```

	Store	Dept	Date	Weekly_Sales	IsHoliday	Type	Size	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	CPI	Unemployment
0	1	1	2010-02-05	24924.50	False	A	151315	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106
1	1	2	2010-02-05	50605.27	False	A	151315	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106
2	1	3	2010-02-05	13740.12	False	A	151315	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106
3	1	4	2010-02-05	39954.04	False	A	151315	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106
4	1	5	2010-02-05	32229.38	False	A	151315	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106

Table 4: First 5 rows of merged train.csv

Exploratory Data Analysis (EDA)

Descriptive Statistics:

Based on the descriptive statistic table generated below (Table 5), most of the numeric attributes look reasonable. Size, temperature, fuel price, CPI, and unemployment rate do not have negative numbers and do not have missing data. It is fair to have negative Weekly_Sales as some people return the goods for a refund. Whereas for Markdown 1 to 5, more than 200k to 300k data are missing, and this is because Markdown 1 to 5 would only record when there are promotional events happen in particular stores of Walmart (Fig.6).

	Weekly_Sales	Size	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	CPI	Unemployment
count	421570.00	421570.00	421570.00	421570.00	421570.00	421570.00	421570.00	421570.00	421570.00	421570.00	421570.00
mean	15981.26	136727.92	60.09	3.36	2590.07	879.97	468.09	1083.13	1662.77	171.20	7.96
std	22711.18	60980.58	18.45	0.46	6052.39	5084.54	5528.87	3894.53	4207.63	39.16	1.86
min	-4988.94	34875.00	-2.06	2.47	0.00	-265.76	-29.10	0.00	0.00	126.06	3.88
25%	2079.65	93638.00	46.68	2.93	0.00	0.00	0.00	0.00	0.00	132.02	6.89
50%	7612.03	140167.00	62.09	3.45	0.00	0.00	0.00	0.00	0.00	182.32	7.87
75%	20205.85	202505.00	74.28	3.74	2809.05	2.20	4.54	425.29	2168.04	212.42	8.57
max	693099.36	219622.00	100.14	4.47	88646.76	104519.54	141630.61	67474.85	108519.28	227.23	14.31

Table 5: Descriptive statistic for the numeric attributes

```
Store              0
Dept               0
Date               0
Weekly_Sales       0
IsHoliday          0
Type               0
Size               0
Temperature        0
Fuel_Price         0
MarkDown1     270889
MarkDown2     310322
MarkDown3     284479
MarkDown4     286603
MarkDown5     270138
CPI                0
Unemployment       0
dtype: int64
```

Fig.6: NAs in the merged dataset

Outliers:

By running boxplot (Fig. 7), it is noticeable that seven attributes have many outliers, including Weekly_Sales, Markdown 1 to 5, and the unemployment rate. No outliers have removed as it may contain valuable information for the analysis.

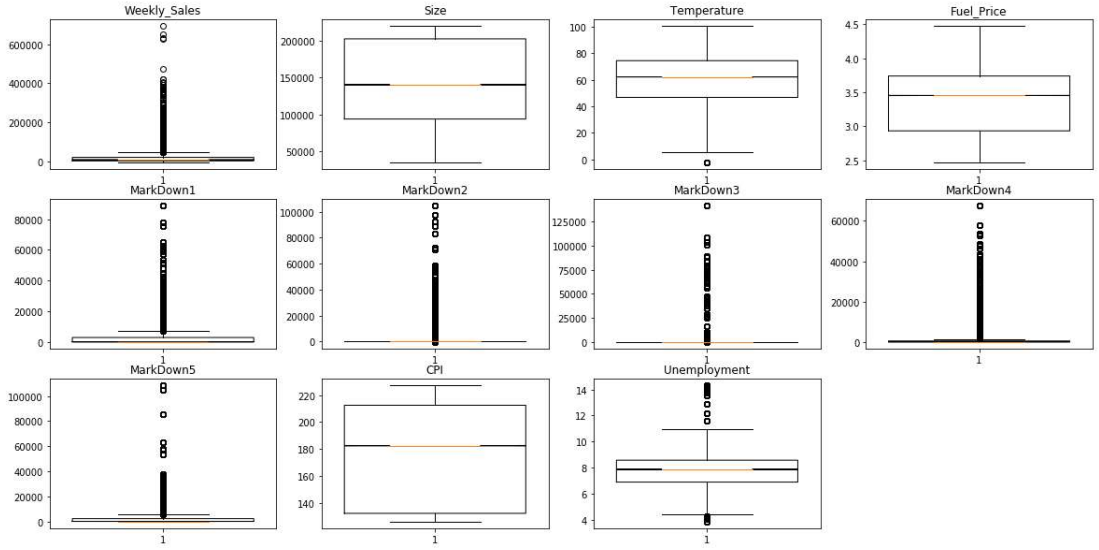

Fig.7: Outliers

Top and Bottom Five Stores in Average Sales:

By calculating the average weekly sales of each store, the highest contributors are store number 20, 4, 14, 13, and 2, each of these stores can generate more than $25,000 each week during 2010 to 2012 (Fig.8a). Those five stores total sales accounted for more than 22% of the total sales for 2010 to 2012 (Table 8b). Whereas store numbers 5, 33, 44, 3, and 38 contributed $5,000 to $7,000 each week only (Fig. 8a), adding all those bottoms five stores contributed around 4% of 2010 to 2012 (Table 8b).

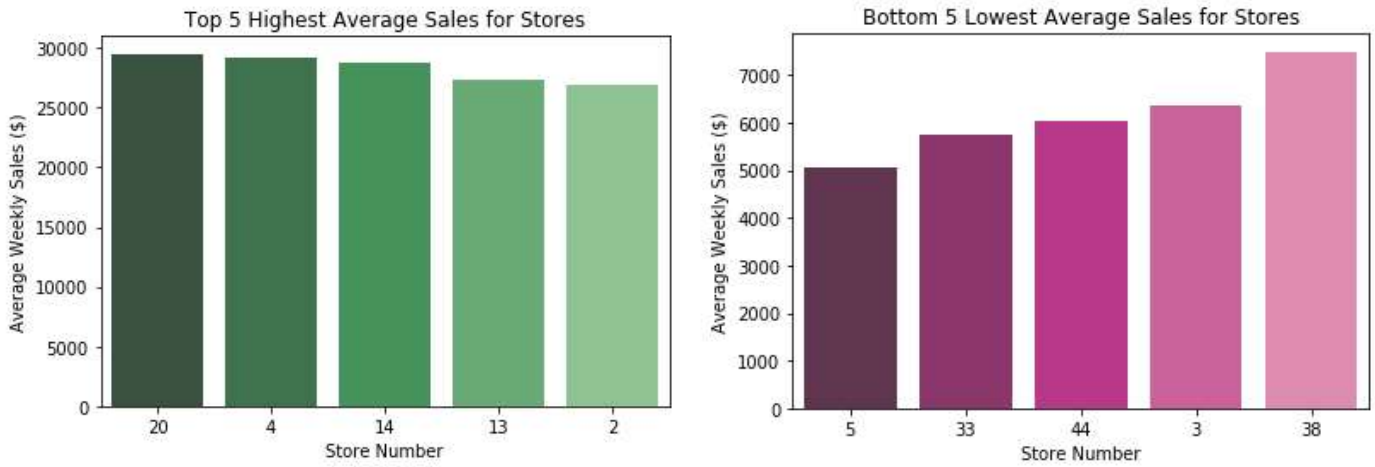

Fig.8a: Top and bottom five stores in sales

Top Store	Store Total Sales	Proportion to Total sales	Top Store	Store Total Sales	Proportion to Total sales
20	301,397,792	4.47%	5	45,475,689	0.67%
4	299,543,953	4.45%	33	37,160,222	0.55%
14	288,999,911	4.29%	44	43,293,088	0.64%
13	286,517,704	4.25%	3	57,586,735	0.85%
2	275,382,441	4.09%	38	55,159,626	0.82%
Sum	1,451,841,802	22%	Sum	238,675,360	4%
Total Sales of 2010-2012:		6,737,218,987			

Table 8b: Store total sales contribution

Top and Bottom Five Departments in Average Sales:

From the perspective of sales in departments, the top five departments are 92, 95, 38, 72, and 65, making average sales of more than $40,000 each week (Fig. 9). To investigate further, based on a number list provided by Walmart[2]The department names for the top five departments for 92, 95, 38, 72, and 65 are Grocery, DSD Grocery, Pharmacy RX, Electronics, and Gasoline, respectively. Whereas, the bottom five departments are 51, 39, 78, 43, 47, each with less than $25 a week. Out of the five departments, only department 39 is on the list, and the name is customer service. The rest would be specialized departments that are not shared. Overall, it is reasonable that the respective departments are the highest sales and lowest sales based on nature.

[2] https://blog.8thandwalton.com/wp-content/uploads/2017/10/Walmart-Department-Numbers.pdf?utm_campaign=Lead%20Generation&utm_medium=email&_hsenc=p2ANqtz--CCzb0keOFMueCFNOrvAAwqcC6gIgcCKebbFQlwPKWD6uCd4srCUwVXlUy64gBJpKUrxKF1DWppwa_q2yszjHxto7NZQ&_hsmi=59683228&utm_content=59683228&utm_source=hs_automation&hsCtaTracking=cdf6bba2-a9f4-4034-8411-9d510d02dc99%7C04f3a820-62fb-4901-9bb1-3b3d0f8161ec

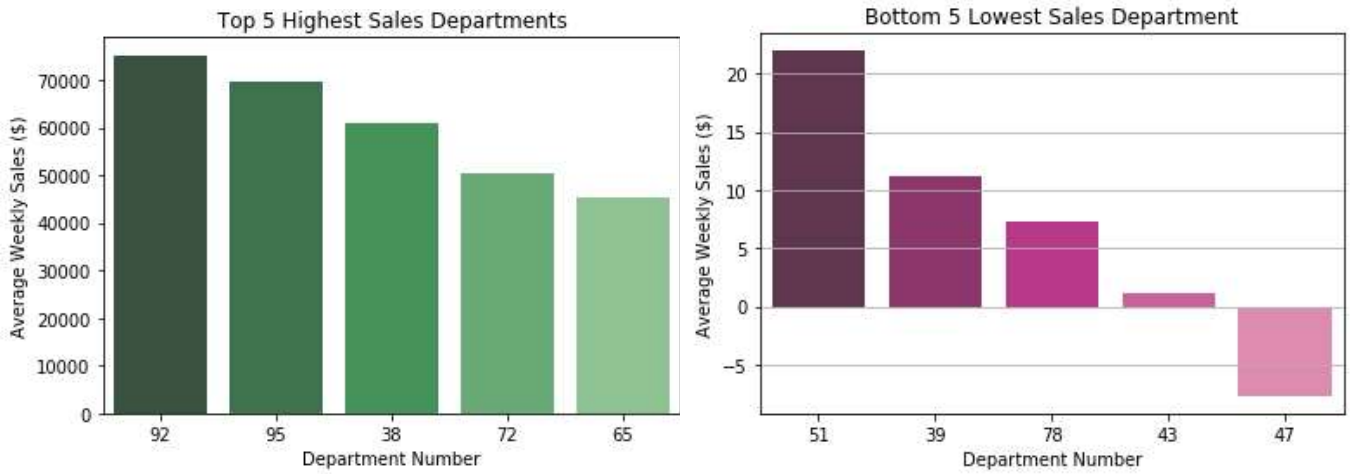

Fig.9: Top and Bottom Five Departments in Average Sales

Comparison of Stores and Departments in Average Weekly Sales

By creating a heat map between stores and departments, it would be easier to visualize the relationships and respective average weekly sales performance (Fig.10). Departments of 38, 40, and 88 to 95 generated the most sales.

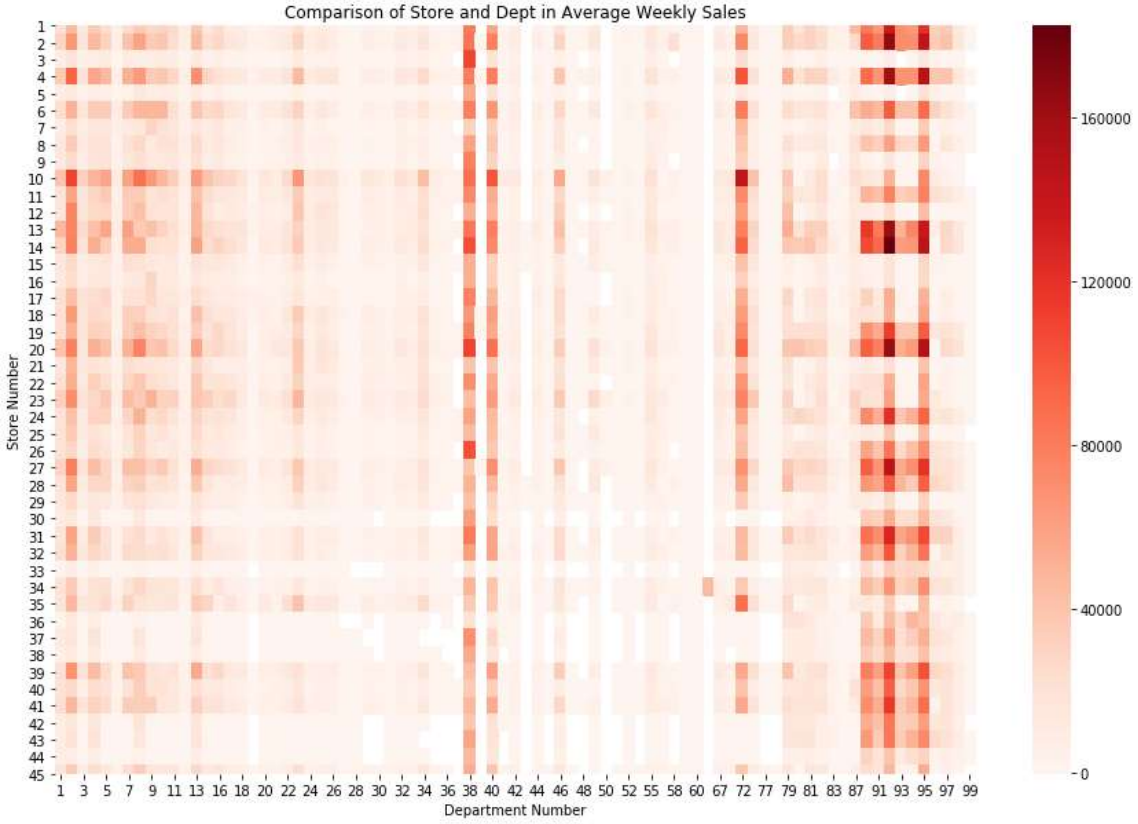

Fig.10: Comparison of stores and departments on average weekly sales

Weekly Sales Comparison for the Dataset (2010,2011,2012)

A summarizing all store and department into one single point for each week. The below figure shows that all three years of average weekly sales are very similar (Fig.11). Especially when there are holidays with Super Bowl, Labor Day, and Thanksgiving, the sales increase dramatically except Christmas. The reason is that people would buy gifts for boxing day before the day after Christmas Day (holiday). Also, it is noticeable that Walmart did not provide other holidays, such as Easter Holiday for week 13 or 14, which also have a high sale.

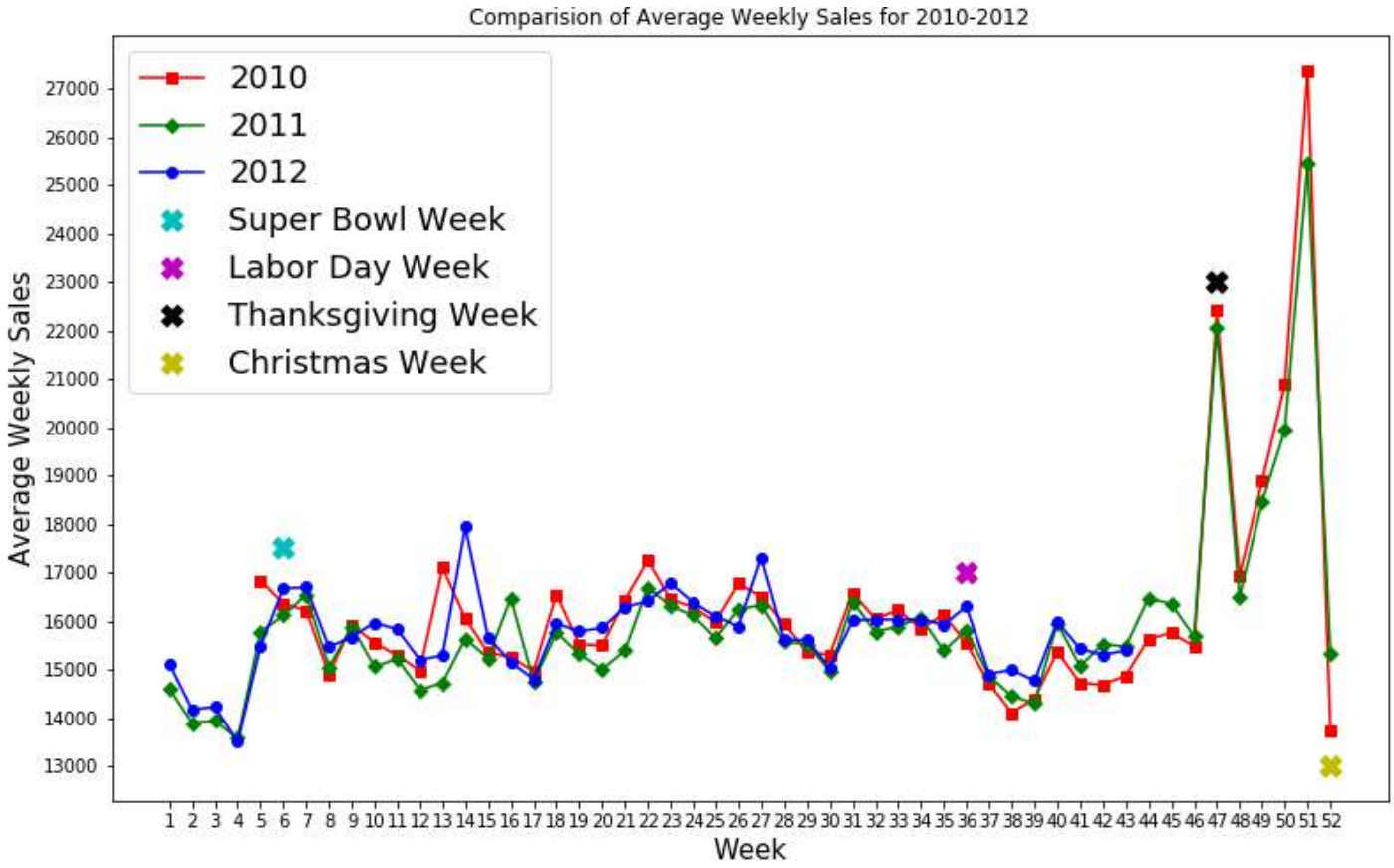

Fig.11: Comparison of average weekly sales for 2010-2012

Comparing the attribute of IsHoliday and Weekly_Sales:

When comparing the attribute of IsHoliday and Weekly_Sales, based on the figure below (Fig.12), weeks that have holidays are indeed generating higher sales. Having a closer look at the box plot, though, seems insignificant, the week that is considered as holiday still has slighter higher sales. Also, based on Fig. 10, week number 52 is a special week. For better prediction, instead of marking week 52 is a holiday week, week 51 should be marked as a holiday.

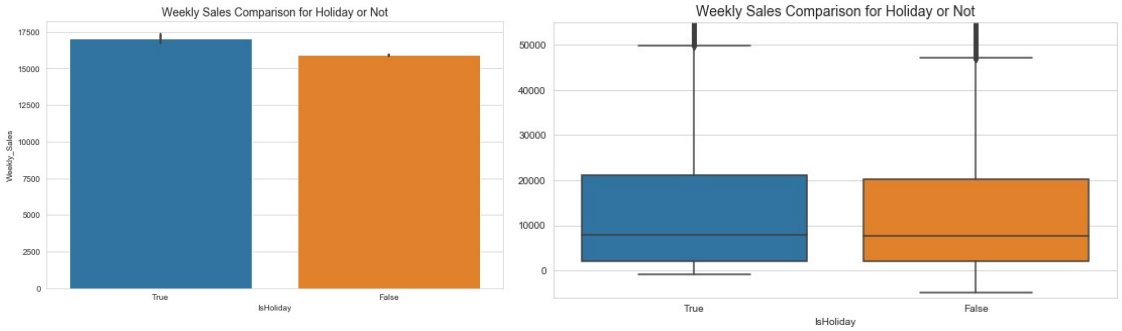

Fig.12: Weekly comparison for IsHoliday or not

Comparing the Attribute Type and Store:

Walmart did not provide information about the attribute "Type", but based on the total 45 stores, around 49% of stores classified as type A, 38% classified as type B; and 13% for type C (Fig.13a). From Fig.13b, it is clear that stores that labeled as type “A” generated the highest average weekly sales, based on the mean. Type "B" generated lesser sales compared to "A", but higher than "C". To conclude, the order of most top weekly sales for types is A> B> C.

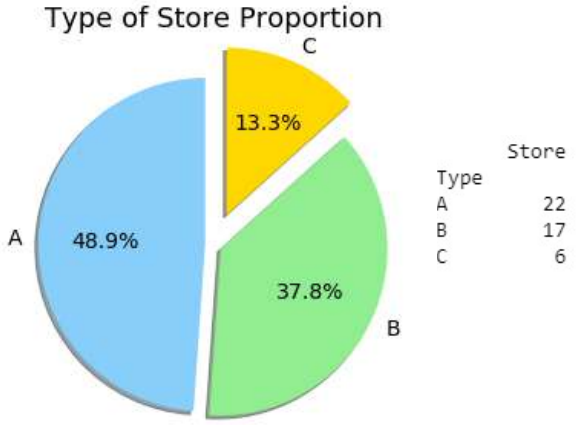

Fig.13a: Type of store proportion

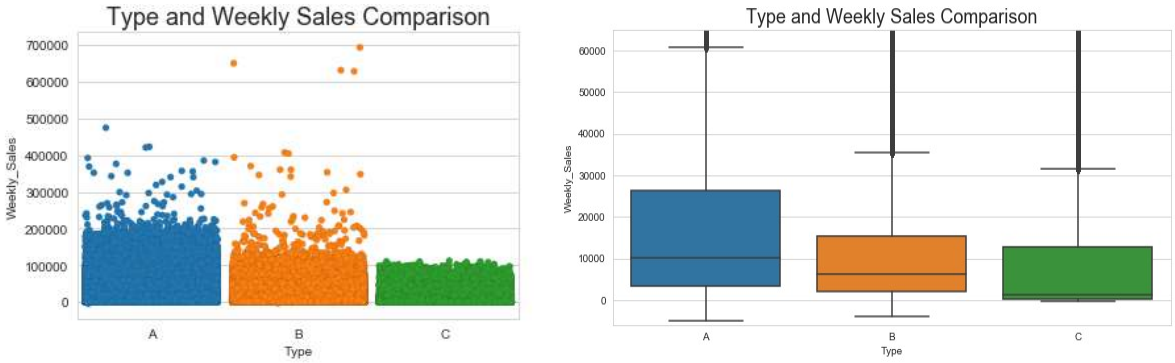

Fig.13b: Type and Weekly Sales Comparison

Type and Size Comparison:

Walmart may also label stores based on their size. Based on the below figure (Fig.14), disregard for some of the outliers, it is clear that type "A" store is the biggest, with a mean of more than 200,000 square feet. Then type "B", with an average of 110,000 squared feet. Lastly, type "C", that are less than 50,000 squared feet.

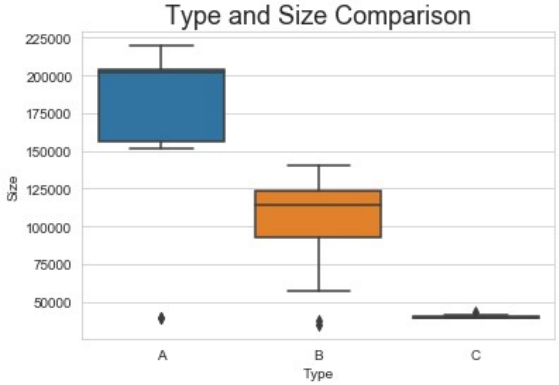

Fig.14 Type and size comparison

Data Cleaning and Preprocess

Based on the previous analysis of the dependent variables and independent variables, the following section would be cleaning the data, including the following procedures.

Adding New Attribute of “Year” and “Week”

Adding the attribute of "Year" and "Week" would make it easier to separate the data set on year or week and for visualization.

Replacing “Type” outputs of A, B, C to 3, 2, 1

In the previous section, store label A indicates higher sales than B, then C. Therefore, higher sales stored that are labeled as A would be replaced with 3, B would be replaced as 2, and C would be replaced as 1 for easier comparison in correlations heatmap in the later section.

Replacing “IsHoliday” from Boolean Values to Numeric

By replacing the true and false values with 1 and 0, the data would be more accessible to model and present.

Replacing value of “IsHoliday” Week 51 and Week 52 to 1 and 0 Respectively

As in the previous section mentioned, week 51 and 52 “IsHoliday” value would be switched around, as Christmas causes the sales, but people purchase the gift before the holiday week.

Making the "Date" as Index

For easier management for splitting the dataset into train and test set for time series and regression.

Visualizing the Correlations of the Attributes and Removing Attributes

In the figure below (Fig.15), all of the attributes are compared against the “Weekly_Sales" attribute, as this is the dependent variable. Independent variables such as "Temperature", "Fuel_Price”, “CPI” and “Unemployment” have very low correlation with “Weekly_Sales". Therefore it would be removed. Markdown 1-5 would also be removed as there are many missing values and low correlation as well. Even though the variable of “IsHoliday" has low correlations to weekly sales, it would not be removed as it is an essential attribute for classifying which week has holidayed. Same for "Store", "Dept", "Year" and "Week", those attributes are essential for separating the data set and would not be removed. Lastly, the attribute of "Type" and “Size" has a weak positive linear relationship to Weekly Sales. Therefore, it would not be removed.

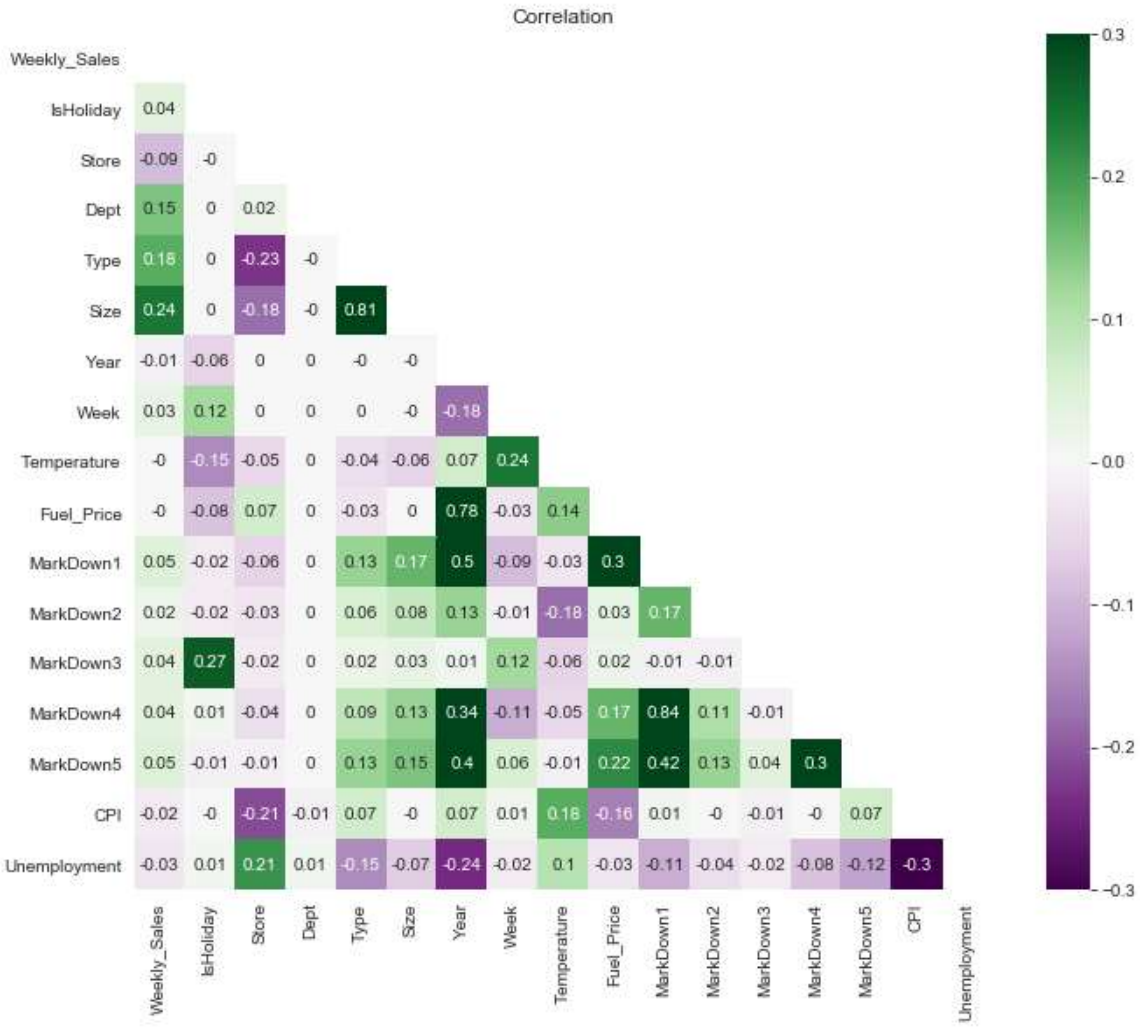

Fig.15: Correlation heatmap

Final Output for Modeling:

Below is the preview of the final output CSV for modeling, only eight attributes remain.

Date	Store	Dept	Weekly_Sales	IsHoliday	Type	Size	Year	Week
2010-02-05	1	1	24924.50	0	3	151315	2010	5
2010-02-05	1	2	50605.27	0	3	151315	2010	5
2010-02-05	1	3	13740.12	0	3	151315	2010	5
2010-02-05	1	4	39954.04	0	3	151315	2010	5
2010-02-05	1	5	32229.38	0	3	151315	2010	5

Fig.16: Final output CSV for modeling

Splitting Dataset

The dataset from the "train.csv" split into train: test and train: cross-validate sets in a ratio of 70:30 and would be arranged based on the date as shown below.

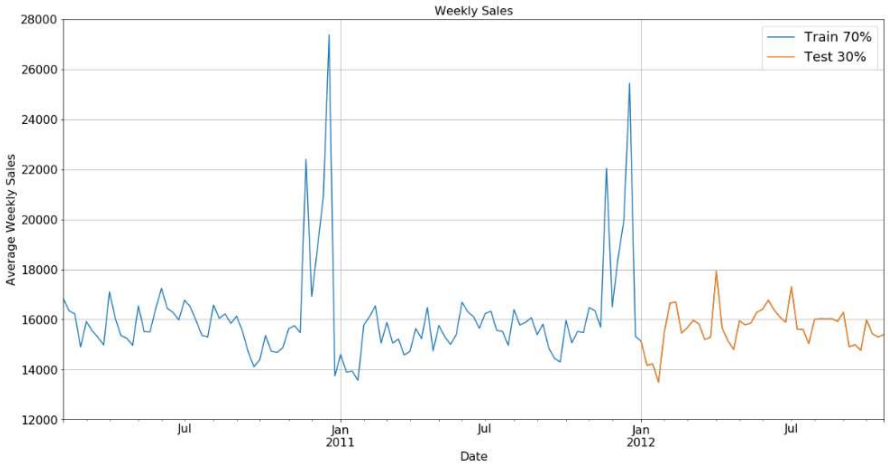

Fig.17: Split Training and Test set

Data Modeling – Time Series Approach:

As a reminder, figure 17 shows the average of all stores and departments. It would not be possible to visualize every single store and department with one graph. Rather than implement all models one by one to each specific store, a more general approach has used. The procedure below for time series and regressions would first use the mean value of each week (combining stores and departments) and then evaluate which methods work best. Lastly, a single department of a store would be used for the final prediction.

The primary purpose of the research paper is to predict further sales for a specific date, store, departments, and provided the data in a time-series format. Therefore, popular time series forecasting methods would be implemented, including AR, MA, ARIMA, SARIMA, and HW.

Decomposition of the Dataset:

Before developing the models, the decomposition of the dataset will be examined first (Fig.18). The first graph above represented the original dataset, which is the same as figure 17. The second graph shows the trend of the dataset, and it captures the slowly moving overall level of the data. Since it goes up and down, there is no clear trend for the data. The third graph shows the seasonality, and it captures a repeated pattern. Here, it is clear that the data has seasonality. This is because near year-end in 2010 and 2011 have higher sale values, and it repeats. Lastly, the last graph of residual is the subtraction of seasonal and trend, which are not captured by both trend or seasonality. All of the residuals' values are not significant compared to the dataset, and a lot of the residuals align to the center.

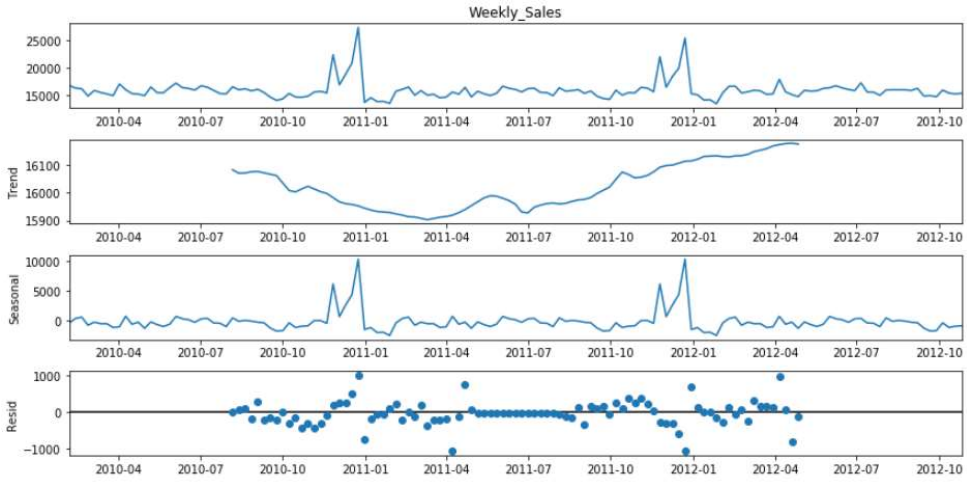

Fig.18: Decomposition of the dataset

The time series model assumes that the data are stationary, meaning that it has a constant mean, variance, and covariance. If the dataset is not stationary, adjustments such as differentiation or transformation are needed. Augmented Dickey-Fuller Test used to test stationarity.

Augmented Dickey-Fuller Test (ADF):

To test whether the data are stationary, the ADF test used to test stationarity. ADF is a method used to check whether the data has a unit root present in the time series data; it also included the lagged terms for determination. It tests the dataset using a hypothesis. H0 suggests unit root presented, whereas H1 indicates that the data is stationary. After running the ADF, it will provide ADF statistics and a p-value. The value of the ADF statistic should be negative, and the more negative the number is, the higher the chance of rejecting H0.

The below figure shows the ADF statics and p-value after running (Fig.19). The ADF statistic is -5.93, and the p-value is near zero. For the research paper, a 95% confidence level was set. Since the p-value is lesser than 5%, we rejected H0, meaning that the time series is stationary and could be applied to AR, MA, ARIMA, SARIMA model.

```
Augmented Dickey Fuller Test to Test Stationary
ADF Statistic: -5.930803
p-value: 0.000000238
Critical Values:
        1%: -3.479
        5%: -2.883
        10%: -2.578
Reject H0, Time Series is Stationary
```

Fig.19: ADF test

Auto Regressor (AR):

The data would first be predicted with AR, which is similar to the linear regression model and predicts value based on a linear combination of past input values. Even though this method is simple and would not be a good predictor for the sales prediction, it is still worth looking before implementing the whole ARIMA and SARIMA.

In the below figure (Fig.20), after implementing the AR model from the train set to the test set, the prediction is shown in the red line. Further investigation by using a performance metric would be in the comparison session.

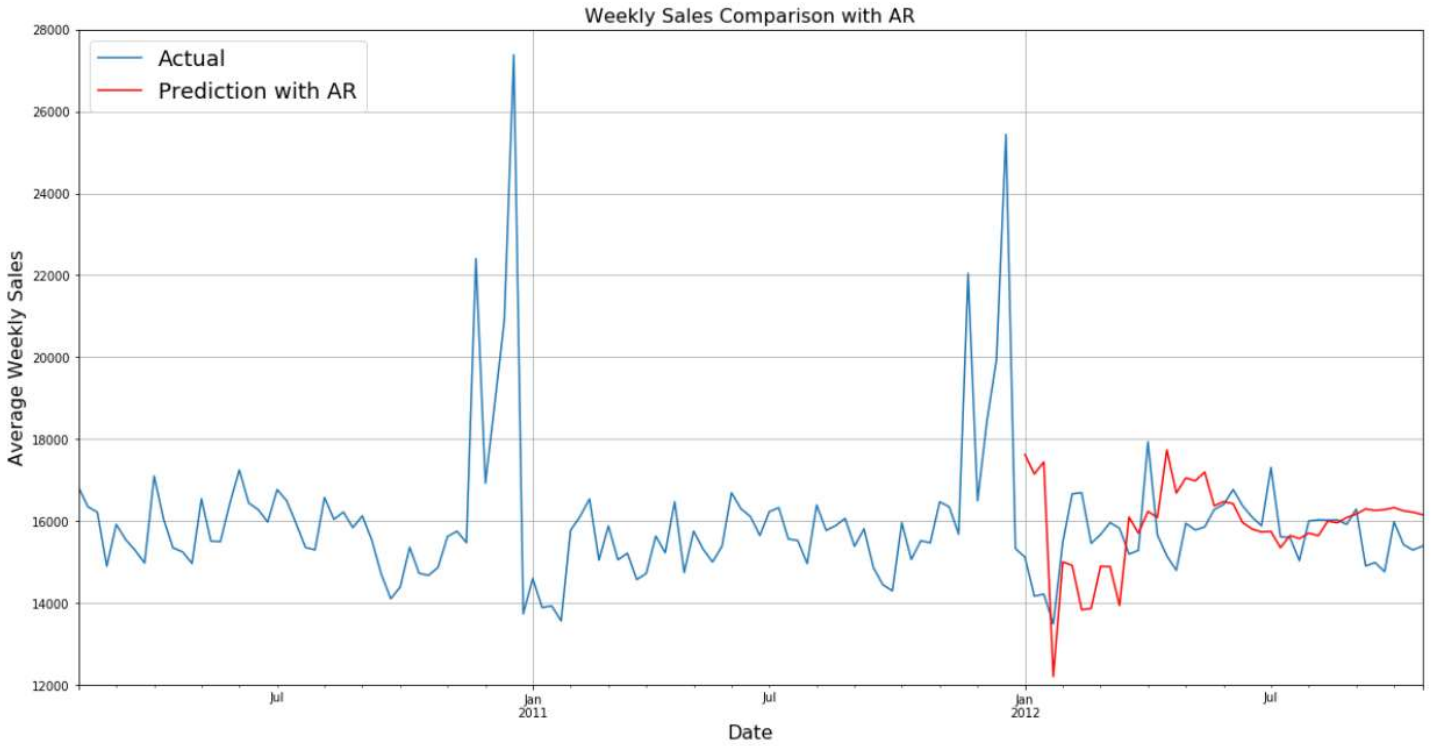

Fig.20: AR prediction compared to the actual data

Moving Average (MA)

Like MA, it is a simple model and would not be an excellent model to predict, but it is still a good model as a baseline to ARIMA and SARIMA. The difference of MA to AR is that rather than using past values, MA uses previous forecast error to predict value. Below is the prediction of MA (Fig.21).

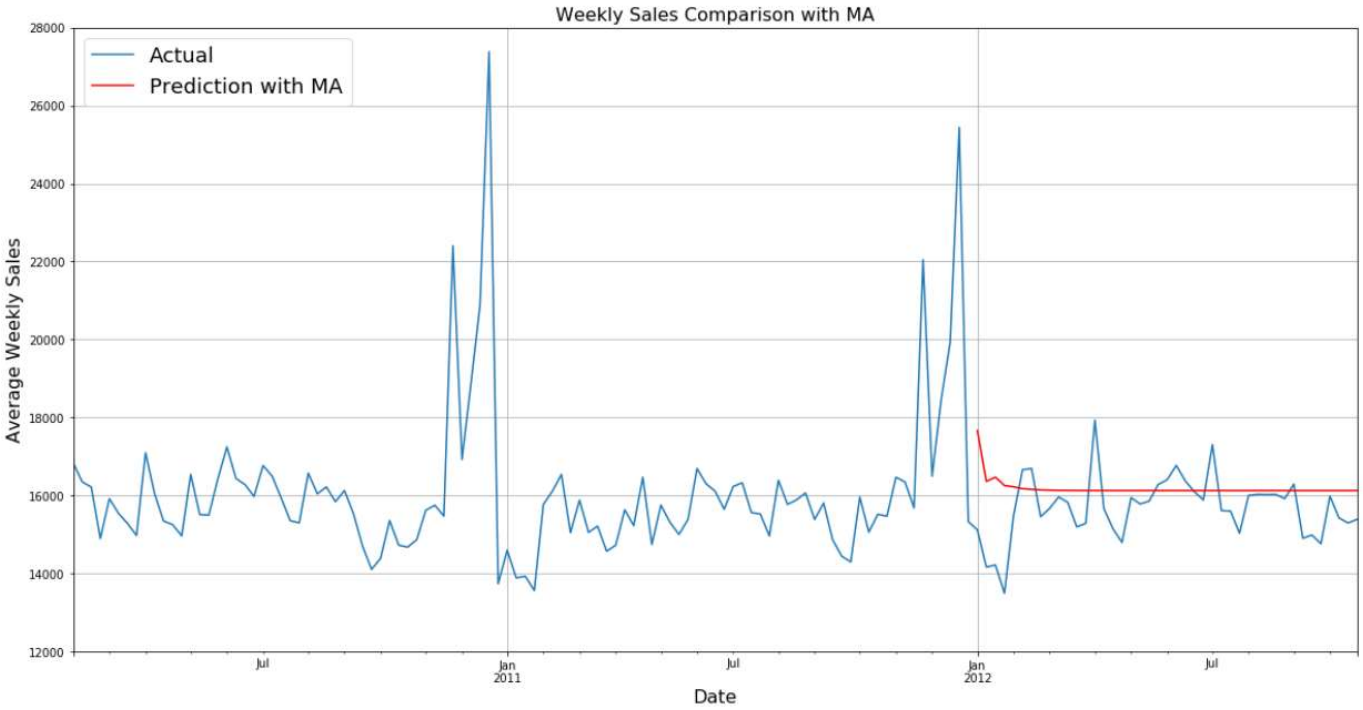

Fig.21: MA prediction compared to the actual data

Auto-Regressive Integrated Moving Average (ARIMA):

ARIMA is similar to AR and MA as it uses the combination of the parameters. The most crucial part is to find the appropriate values for "p,d,q". The value of p is the number of Auto Regressors (AR), where d is the difference (I), and q is the number of Moving Average (MA).

To hyper tune, the number for the "p, d, q", one method is to plot Auto Correlated Function (PCF) and Partial Auto Correlated Function (PACF) graph and pick the numbers. Another technique, which used in this research paper, is to use a grid search function to test the combination values of "p, d, q" and choose the best combination based on Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC).

Figure 22 shows the final result of the function used to select the best combination of "p, d, q", which is "2, 0, 0" respectively, as it has the lowest average AIC and BIC value. Figure 23 shows the diagnostics of the current parameters with residual plots. The Top left graph shows that the residual errors seem to fluctuate near zero except for sales near the year. The top right suggests the residuals are almost normal distribution with a mean near zero. The bottom left shows that the distribution is skew as some data are way off the red line. The bottom right ACF plot shows that the residual errors are not autocorrelated; this current model seems able to explain all patterns. Figure 24 shows the combination of p=2, d=0, q=0, and how the prediction looks compared to the actual dataset. Compared to the AR model earlier, the ARIMA provided a straight line that would not predict year-end sales, but further investigation by using performance metrics would be in the comparison session.

```
Fit ARIMA(2,0,0)x(0,0,0,0) [intercept=True]; AIC=1803.956, BIC=1814.376, Time=0.142 seconds
Fit ARIMA(2,0,1)x(0,0,0,0) [intercept=True]; AIC=1805.975, BIC=1819.001, Time=0.043 seconds
Fit ARIMA(2,0,2)x(0,0,0,0) [intercept=True]; AIC=1804.562, BIC=1820.193, Time=0.209 seconds
Total fit time: 0.542 seconds
                                SARIMAX Results
==========================================================================================
Dep. Variable:                          y   No. Observations:                  100
Model:               SARIMAX(2, 0, 0)   Log Likelihood                -897.978
Date:                Mon, 27 Jul 2020   AIC                           1803.956
Time:                        12:54:21   BIC                           1814.376
Sample:                             0   HQIC                          1808.173
                                  - 100
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept   9009.8639   1527.552      5.898      0.000    6015.918    1.2e+04
ar.L1          0.2510      0.060      4.190      0.000       0.134       0.368
ar.L2          0.1903      0.098      1.949      0.051      -0.001       0.382
sigma2      3.756e+06      3.236   1.16e+06      0.000    3.76e+06    3.76e+06
===================================================================================
Ljung-Box (Q):                       28.60   Jarque-Bera (JB):               485.37
Prob(Q):                              0.91   Prob(JB):                         0.00
Heteroskedasticity (H):               9.27   Skew:                             2.27
Prob(H) (two-sided):                  0.00   Kurtosis:                        12.79
===================================================================================
```

Fig.22: Auto selector result

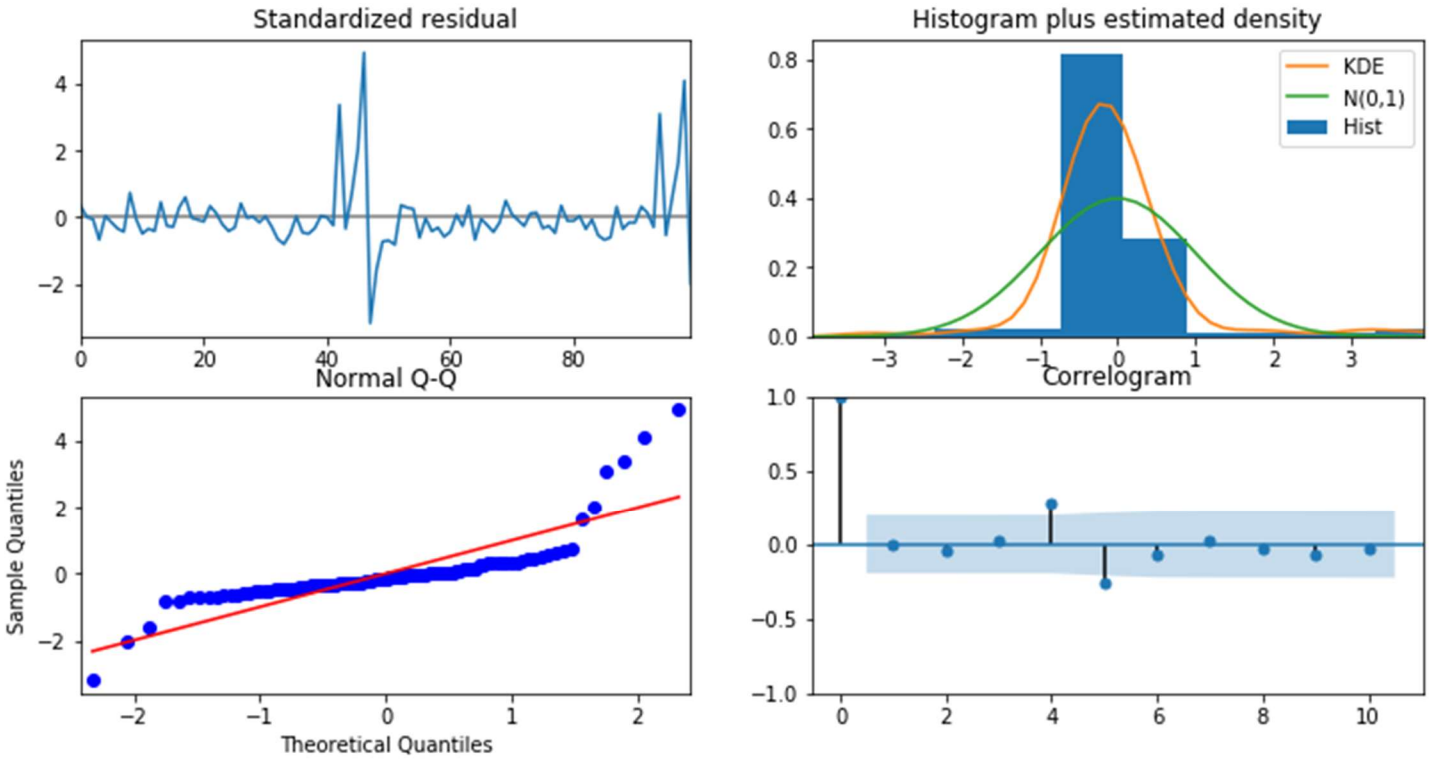

Fig.23 Diagnostics of the ARIMA (2, 0, 0)

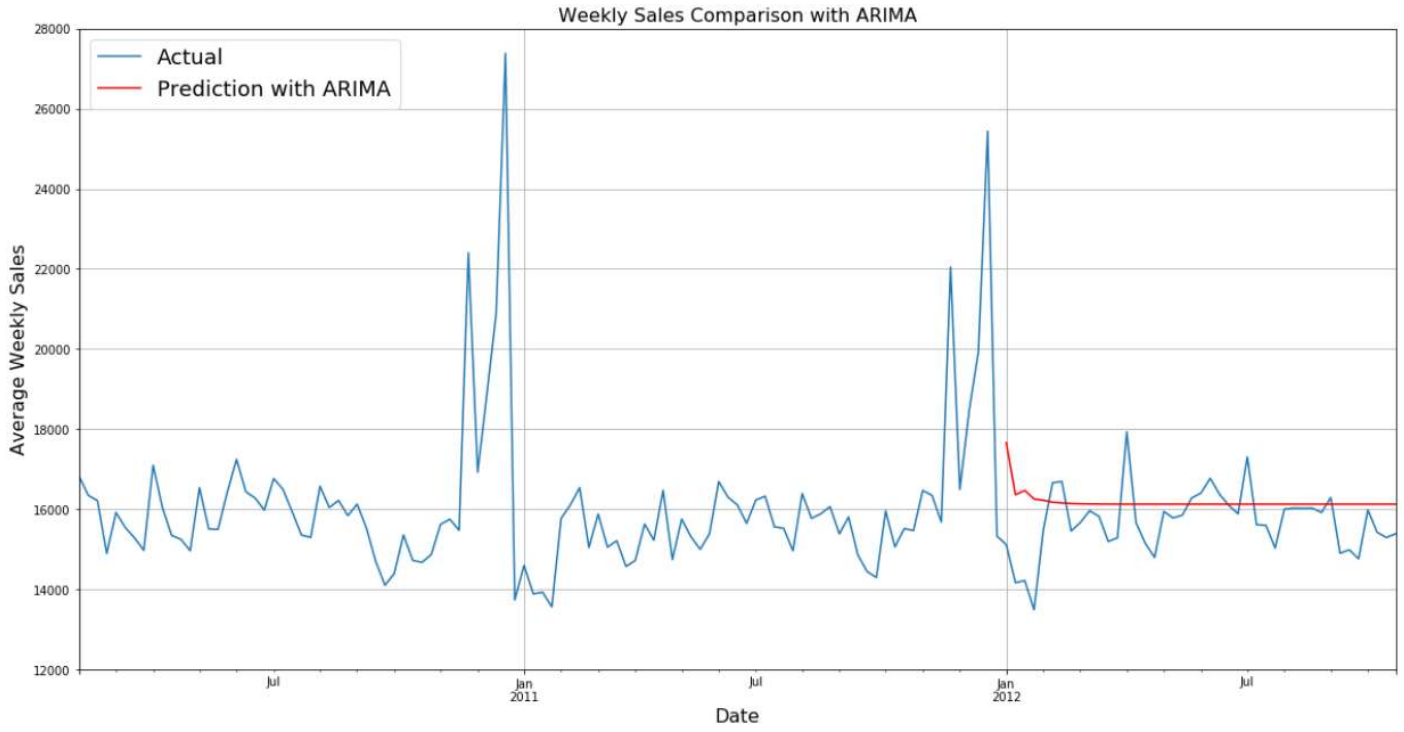

Fig.24: ARIMA compared to actual data

Seasonal Auto-Regressive Integrated Moving Average (SARIMA):

Again, similar to AR, MA, and ARIMA, the SARIMA model included the seasonal element, which would further increase accuracy. Additional to the parameters of p, d, and q, there are four more elements for the SARIMA, which are "P", "D", "Q", and "s". The "P" is the number of autoregressive terms of lags for the time series. The "D", is the differencing for stationary time series data. The "Q" is the number of moving average terms or lag of forecast errors. Lastly, "s" is the seasonal length for the data. Figure 25, similar to ARIMA, performed hyper tuning, and the best combination for the data for SARIMA is (1, 0, 2) (1, 0, 1, 52) as it has the lowest average of AIC and BIC.

In the residuals plots (Fig.26), the results are similar to ARIMA (2, 0, 0), but the standardized residual plot shows that the residuals are closer to zero than ARIMA. Also, the histogram and density graph are slightly flatter than ARIMA. The normal Q-Q graph has more residuals closer to the red line compared to the ARIMA. Overall, an improvement compared to ARIMA.

In figure 27, the predicted compared to the actual value looks more promising than the ARIMA, as the predicted values are close to the real value, unlike ARIMA, which is only a straight line. Further investigation by using a performance metric would be in the later session.

```
Fit ARIMA(1,0,1)x(2,0,0,52) [intercept=True]; AIC=1781.455, BIC=1797.086, Time=11.674 seconds
Fit ARIMA(0,0,1)x(2,0,0,52) [intercept=True]; AIC=1797.962, BIC=1810.988, Time=6.083 seconds
Fit ARIMA(2,0,1)x(2,0,0,52) [intercept=True]; AIC=1778.425, BIC=1796.661, Time=24.313 seconds
Total fit time: 200.232 seconds
                                      SARIMAX Results
==========================================================================================
Dep. Variable:                                   y   No. Observations:                  100
Model:             SARIMAX(1, 0, 2)x(1, 0, [1], 52)   Log Likelihood                -858.542
Date:                              Mon, 27 Jul 2020   AIC                           1731.084
Time:                                      12:57:42   BIC                           1749.321
Sample:                                           0   HQIC                          1738.465
                                                - 100
Covariance Type:                                opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept   1.333e+04   5256.459      2.536      0.011    3027.067    2.36e+04
ar.L1         -0.9978      0.074    -13.458      0.000      -1.143      -0.852
ma.L1          1.2562      0.133      9.472      0.000       0.996       1.516
ma.L2          0.2603      0.070      3.744      0.000       0.124       0.397
ar.S.L52       0.5856      0.156      3.760      0.000       0.280       0.891
ma.S.L52       0.7174      0.309      2.323      0.020       0.112       1.323
sigma2      7.304e+05     30.324   2.41e+04      0.000     7.3e+05    7.31e+05
==========================================================================================
Ljung-Box (Q):                       31.82   Jarque-Bera (JB):              1283.06
Prob(Q):                              0.82   Prob(JB):                         0.00
Heteroskedasticity (H):               1.31   Skew:                             2.96
Prob(H) (two-sided):                  0.44   Kurtosis:                        19.52
==========================================================================================
```

Fig.25: Auto selector result

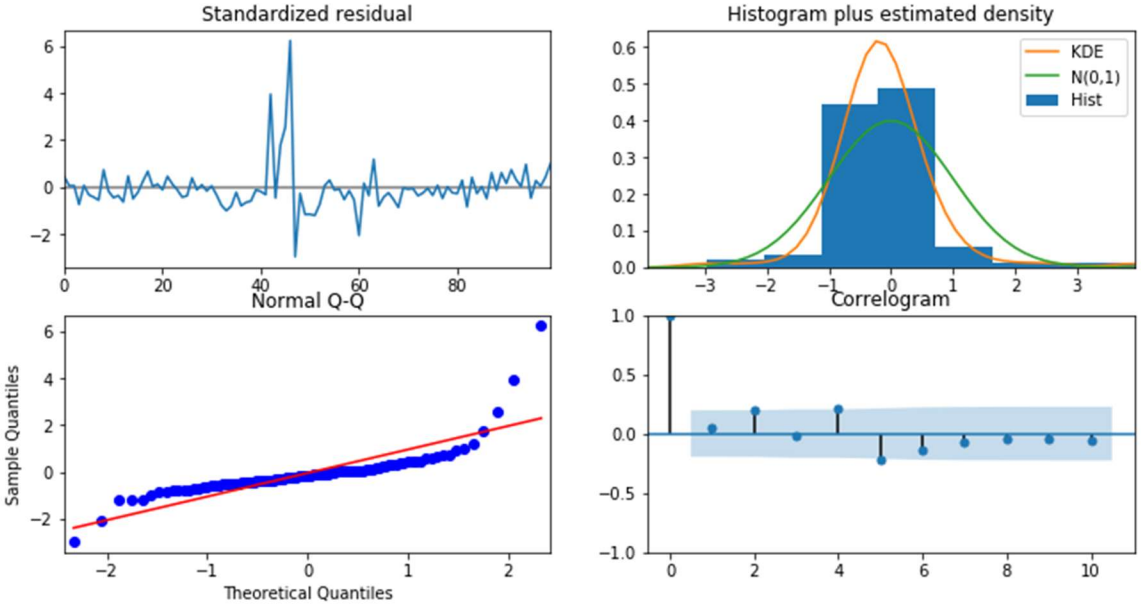

Fig.26: Diagnostics of the SARIMA (1, 0, 2)x(1, 0, 1, 52)

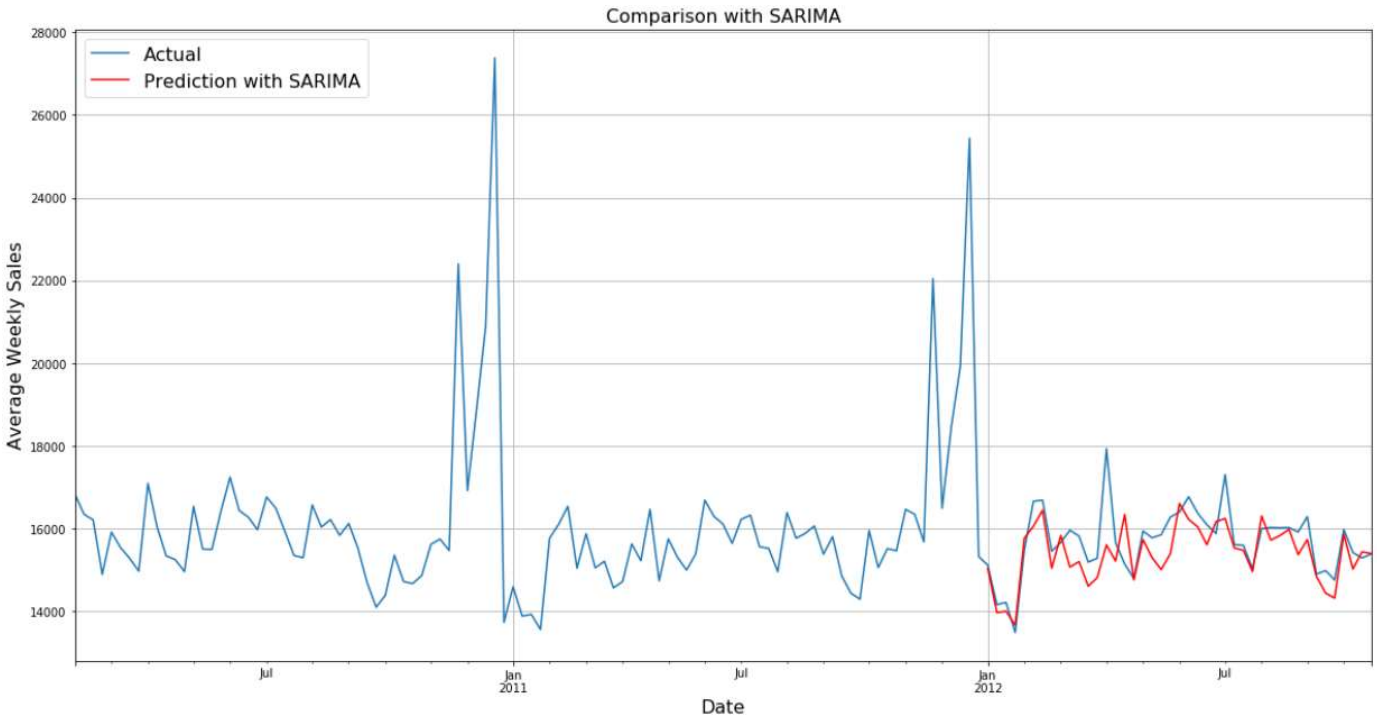

Fig.27: SARIMA compared to actual data

Holt-Winters' Method (HW):

HW is also called Triple Exponential Smoothing algorithm as it smoothes out three parameters: level, trend, and seasonality exponentially. The beauty of the HW package is that the preset settings could already generate a high accuracy prediction (Fig.28), so far, the model fit well as SARIMA. Further investigation by using a performance metric would be in the comparison session.

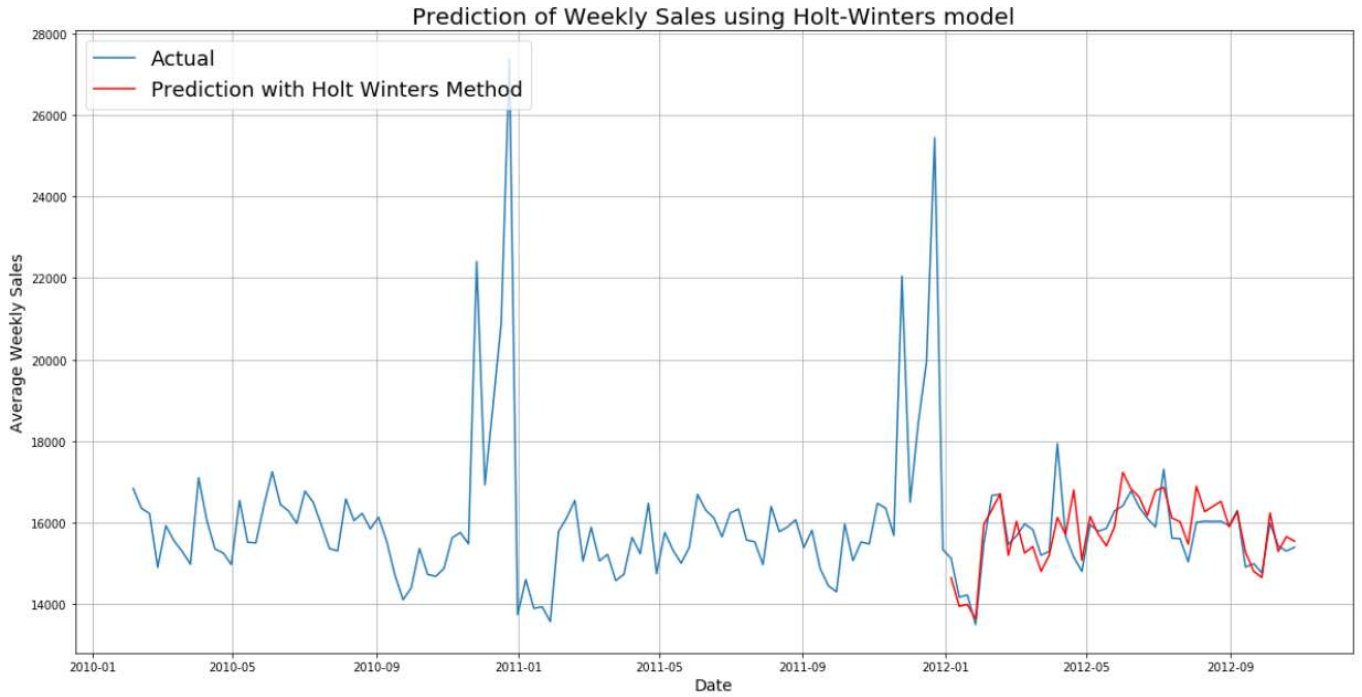

Fig.28: HW compared to actual data

Regression Approach

In the previous section, only time-series approaches have been used and examined as the data set consists of a timestamp. In this section, machine learning regression models would be tested based on attributes relationship. The models included three common methods: Decision Tree, Random Forest Regressor, and XGBoost.

Decision Tree Regressor:

Decision tree regressor is a supervised learning algorithm that the final output is base on comparing different values of predictors against threshold values. This technique constructs a tree model, the tree starts with the topmost node, which is the root node (class), the method tests the data set on the root node, and only two outcomes would be provided the branches. The tree keeps extending until all of the data covered in the decision tree. Similar to time series methods, hyper tuning was performed for the regression approach. RandomizedSearchCV was implemented to determine what parameters are best for decision tree regressors. The reason for not using GridSearchCV is because it a long time to run the model, and the results are similar to RandomizedSearchCV. In this research paper, only “min_samples_leaf”, “min_samples_split”, and “max_depth” are being tested, as these are the most common parameters to alter for decision tree regressor. The range of the parameters are listed in the figure below, the parameters covered low level and high level and the best combination are near the middle of the range: “min_samples_split=54”, “min_samples_leaf=25”, and “max_depth=440” as below (Fig.29).

```
#Define Decision Tree
model = DecisionTreeRegressor(random_state=1)

#Setting parameters to test on Decision Tree
param1 = [
    {'min_samples_leaf': range(1,51,2),
     'min_samples_split':range(2,100,2),
     'max_depth':range(5,1000,5)}
]

#Perform Randomized Search CV
grid_search = RandomizedSearchCV(model, param1, cv = 3, verbose = 3,
                                 n_jobs = -1)
grid_search.fit(X_train, y_train)
results = grid_search.cv_results_
best_param=grid_search.best_params_

#Display the best result
best_param
```

```
Fitting 3 folds for each of 10 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 4 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:    6.5s finished
{'min_samples_split': 54, 'min_samples_leaf': 25, 'max_depth': 440}
```

Fig.29: RandomizedSearchCV for decision tree regressor

Random Forest Regressor:

Like decision tree regressor, the random forest regressor is an ensemble regressor using many decision trees models. Instead of splitting the node base on probability, a random forest regressor randomly selected a subset of variables and split each node. Eventually, when used to predict data, instead of based on one single decision tree, the output would be predicting base on multiple decision trees. Therefore theoretically, a random forest regressor would outperform a decision tree. For hyper turning the parameters for random forest regressor, it is the same as decision tree regressor with an addition of the “n_estimators” parameters, which is the number of decision trees to use. The ranges for the parameters are listed in figure 30. The best combination is: “n_estimators=200”, “min_sample_split=15”, “min_samples_leaf=8”, and “max_depth=200”.

```
#Define Random Forest Regressor
model = RandomForestRegressor(random_state = 42)

#Setting parameters to test on Random Forest Regressor Model
params={
 "n_estimators"     : range(100,300,100) ,
 "max_depth"        : [5,25,50,100,200],
 "min_samples_split": [2,5,8,10,15,20],
 "min_samples_leaf" : [1,2,5,8,10]
}

#Perform Randomized Search CV
grid_search = RandomizedSearchCV(model, params, cv = 3, verbose = 3,
                                 n_jobs = -1)
grid_search.fit(X_train, y_train)
results = grid_search.cv_results_
best_param=grid_search.best_params_

#Display the best result
best_param
```

```
Fitting 3 folds for each of 10 candidates, totalling 30 fits
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 4 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:  7.6min finished
```

```
{'n_estimators': 200,
 'min_samples_split': 15,
 'min_samples_leaf': 8,
 'max_depth': 200}
```

Fig.30: RandomizedSearchCV for random forest regressor

XGBoost:

XGBoost is the short name of eXtreme Gradient Boosting. It is a custom tree building algorithm. It is also similar to a decision tree, but instead, rather than training individual trees and perform prediction, XGBoost train models in succession. Each successive model would be trained and improve previous error until no further improvement can be made. With the decision tree, even it is being repeated. It may end up making the same mistake as they are trained in isolation.

The common parameters for hyper tuning included: “learning_rate”, ”max_depth”, ”min_child_weight”, ”gamma”, and “colsample_bytree”.

The ranges are shown in the figure below and the best combination is: “learning_rate=0.05”, ”max_depth=25”, ”min_child_weight=5”, ”gamma=0.1”, and “colsample_bytree=0.7” (Fig.31).

```
#Define XGBoost
model = xgb.XGBRegressor(random_state = 1)

#Setting parameters to test on XGBoost
params={
 "learning_rate"    : [0.05, 0.10, 0.15, 0.20, 0.25, 0.30 ] ,
 "max_depth"        : [5,25,50,100,200,500,1000],
 "min_child_weight" : [ 1, 3, 5, 7 ],
 "gamma"            : [ 0.0, 0.1, 0.2 , 0.3, 0.4 ],
 "colsample_bytree" : [ 0.3, 0.4, 0.5 , 0.7 ]
}

#Perform Randomized Search CV
grid_search = RandomizedSearchCV(model, params, cv = 3, verbose = 3,n_jobs = -1)
grid_search.fit(X_train, y_train)
results = grid_search.cv_results_
grid_search.best_params_
best_param=grid_search.best_params_
#Display the best result
best_param
```

```
Fitting 3 folds for each of 10 candidates, totalling 30 fits
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 4 concurrent workers.
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:  3.0min finished
```

```
{'min_child_weight': 5,
 'max_depth': 25,
 'learning_rate': 0.05,
 'gamma': 0.1,
 'colsample_bytree': 0.7}
```

Fig.31: RandomizedSearchCV for XGBoost

Model Comparison and Performance Measure

As below table, the models are compared against each other using performance measure of MSE, MAE, RMSE. HW has the lowest error in MAE, MSE, and RMSE, and it is the best model to predict the test set from the train set. Surprisingly, the time series models perform much better than regression models (Fig.32).

Methods\Metics	MAE	MSE	RMSE	Rank
AR	1,019.58	1,786,083.79	1,336.44	5
MA	747.95	1,027,941.54	1,013.87	3/4
ARIMA	747.95	1,027,941.54	1,013.87	3/4
SARIMA	422.64	343,466.02	586.06	2
Holt Winters' Method	383.26	282,655.10	531.65	1
Decision Tree Regressor	5,753.38	127,988,545.76	11,313.20	8
Random Forest Regressor	5,126.75	110,149,330.12	10,495.21	6/7
XGBoost	5,343.32	95,199,340.15	9,757.01	6/7

Table 32: Time Series and Regression method comparison

Sales Prediction

The time series so far is based on a single point for each day, and it is for easy visualization and management to view national wide sells. Using regression methods would be the easiest to implement all those methods would be able to predict based on the attributes of each store and departments. It is not a simple task for time series methods, to predict all 115,064 values in the test.csv, one would need to implement HW repeatedly in each separate store and department. Also, since each store and department weekly sales could be affected by different variables, using a single grid search result and applying it to all stores would not be as accurate. Nevertheless, for a shallow understand and implementation of the technique to the problem. One of the store's departments would be selected. For comparison, all of the above models would be performed against the test set, and later would only use HW to perform prediction of future sales.

Choosing a Store and Department:

The most reasonable way to choose a specific store and department is to select the highest average weekly sales store and select the most top average weekly sales of the department to implementation of the model.

From the EDA section, store 20 shows the highest average weekly sales during the period of 2010-2012 (Fig. 8a). By comparing the average deals of the departments from store 20, the below figure indicates that department 92 (Grocery) has the highest average weekly sales of about $160K (Fig.33). Therefore Store 20 Department 92 (S20_D92) is selected.

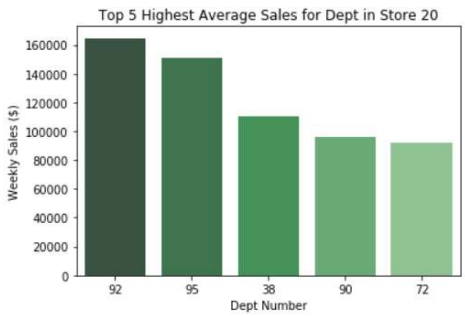

Fig.33: Store 20 top department sales

Compare Regression Models with Train and Test Set for S20_D92

By implementing the regression models with their respective randomized search cv parameters, the results are somewhat disappointing, as shown in figure 34. All of the algorithms are not able to predict the values correctly and are all underestimated. XGBoost seems able to capture the spikes but is all underestimated as well.

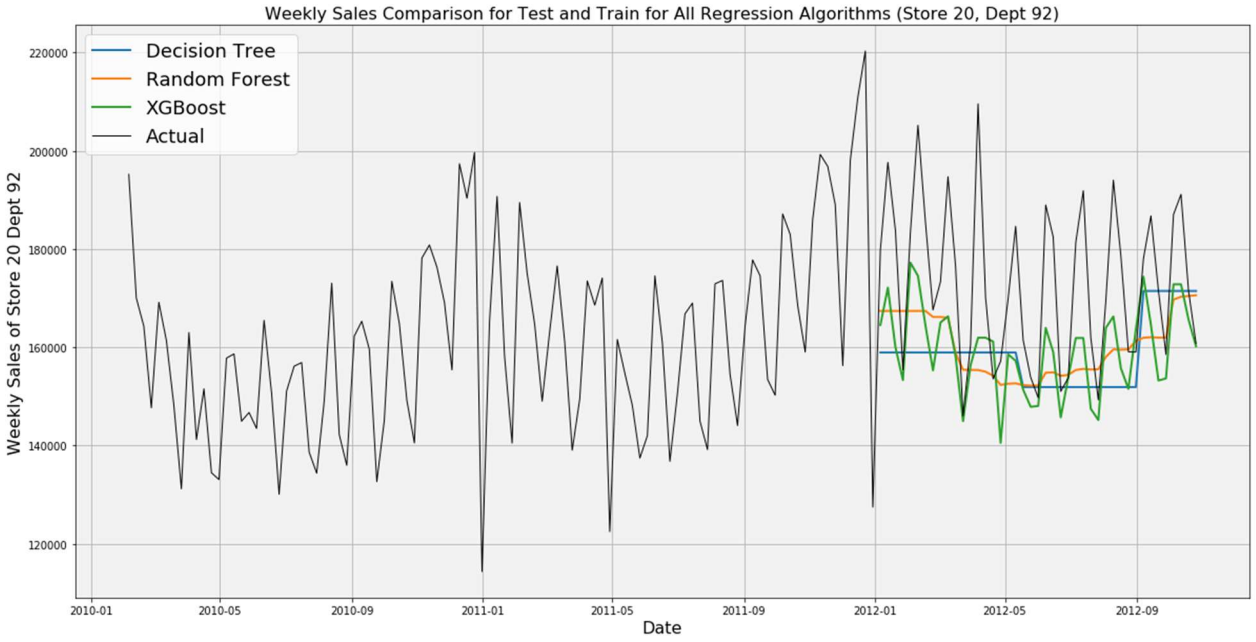

Fig.34: Regression algorithms comparison for train and test set

Compare Time Series Models with Train and Test Set for S20_D92

By contrast, some of the time series models seem promising. In the figure below (Fig.35), AR, SARIMA, and HW appear to capture the highs and low better compared to regression models.

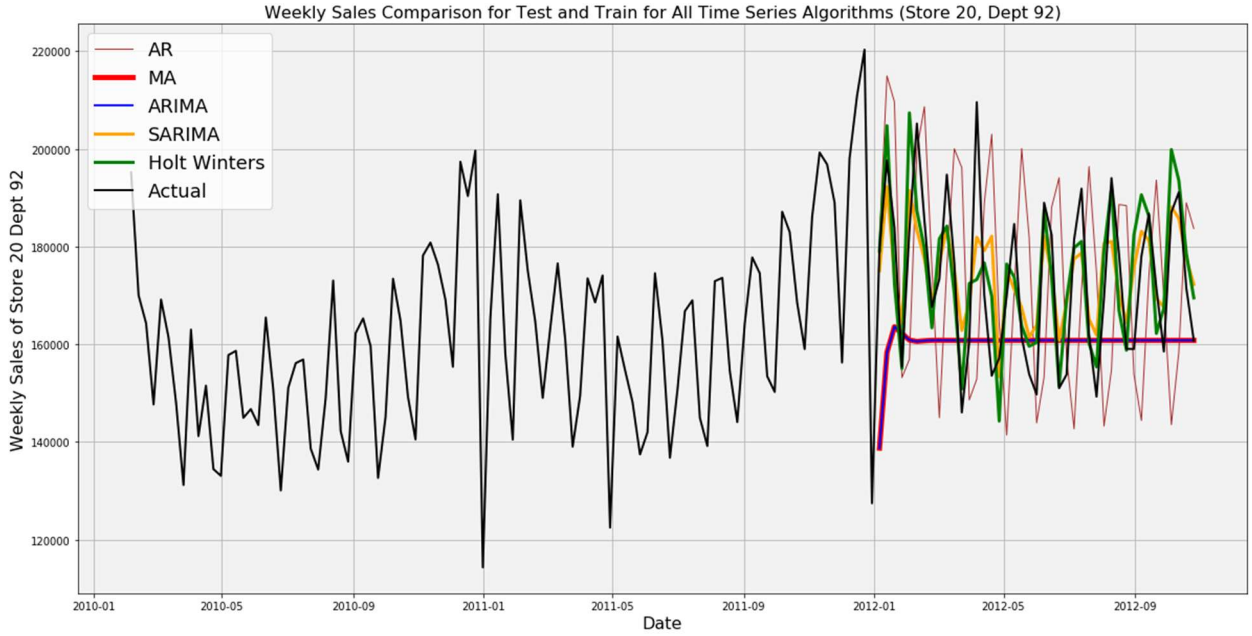

Fig.35: Time series algorithms comparison for train and test set.

For easier visualization, figure 36 shows a slightly closer look at those models. Even though AR seems to be able to capture the highs and low, a lot of the predictions are off by one gap. Also, many of the projections overestimate and underestimate quite a bit, despite the one-week difference. SARIMA and HW, on the other hand, seems able to capture pretty good. HW seems outperformed SARIMA only a slight fraction.

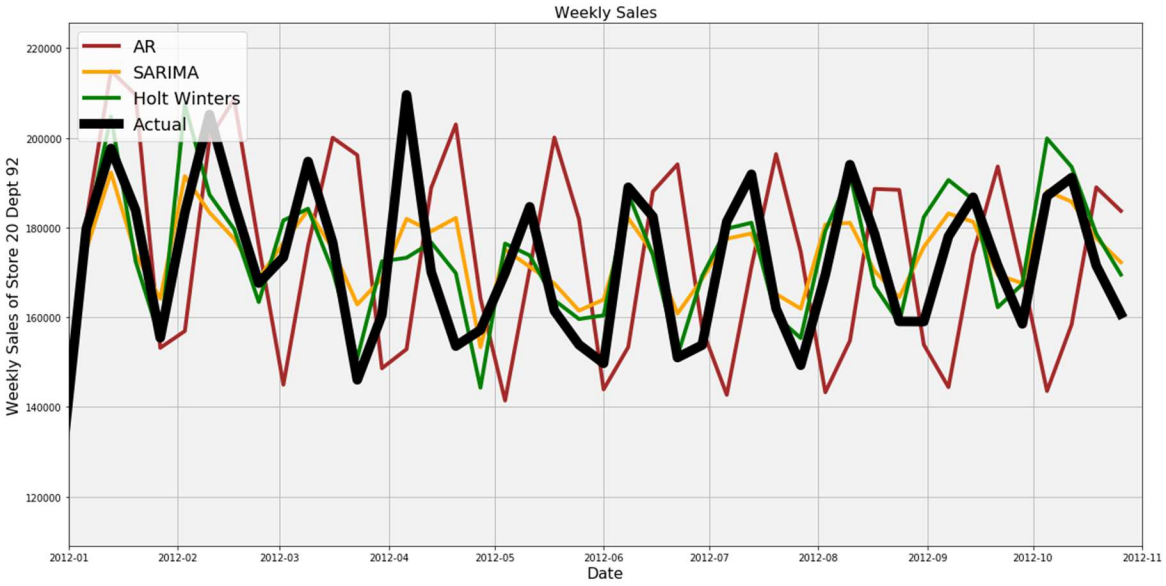

Fig.36: Zoomed version of AR, SARIMA, HW and test set

Sales Prediction for the year of 2012-2013:

In figure 37, the black line represents the original dataset (train.csv), including the train and test set. The blue line represents all the predictions for train and test set from (train.csv), whereas the green line represents the future predictions for late-2012 to mid-2013 (test.csv). The forecast seems pretty reasonable and accurate as of the blue line, and the black line is pretty close together. Additionally, it can predict a slight increase trend from the past two years.

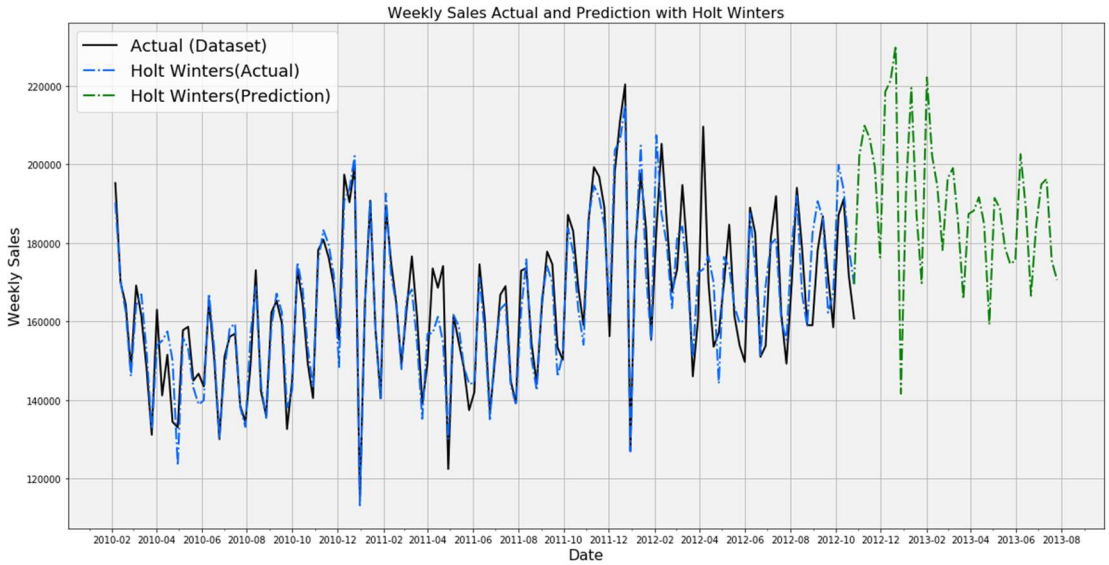

Fig.37: HW Method for S20_D92

Comparison of the Prediction to Previous Year:

When comparing the previous year's weekly sales for S20_D92 (Fig.38) shows that there is an increase in sales overall, the orange line represents 2010, purple line represents 2011. Redline represents 2012, an overall improvement in sales. Therefore, the HW predicts that the sales for late 2012 and mid-2013 would also have an increasing trend, which is reasonable.

Initially, when first look at figure 11, the holiday week seems to affect the weekly store in general. Still, here, the holiday week does not appear to have a significant impact on S20_D92. It may be due to grocery prices not varying, and it is necessary, therefore, not to have a considerable discount. Overall, the HW seems promising and able to capture highs and lows.

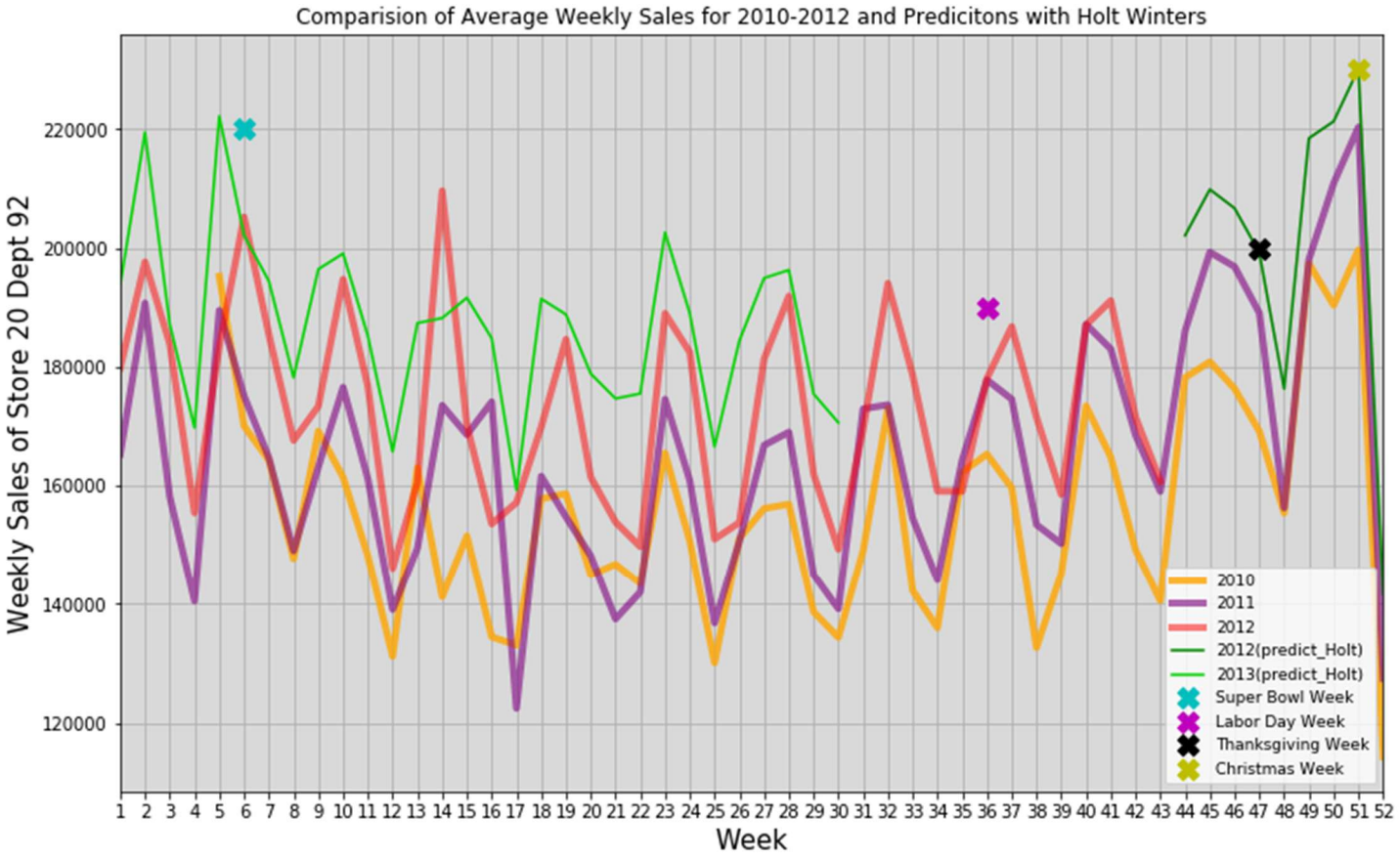

Fig.38: HW prediction for S20_D92 and previous year with holidays

Conclusion

To conclude, based on implementing different time series and regressions models, time series models still performed better than regression models, especially the HW. The reason is that time series models can predict values based on past values, including seasonality and trend. Whereas, regressions models only predict future value based on the currently known attributes and make predictions on those specific attributes. As suggested in the EDA part, many of the attributes do not have strong relations with weekly sales, which may be why the regression models were not able to perform well.

Since the research paper's goal is to have a shallow understanding of both time series models and regression models, some limitations and improvements could be made for the research paper. Firstly, the number of

parameters for hyper tuning could increase but was limited due to the time it would consume. It would not significantly improve the results, especially for regression models. Secondly, the HW could be applied for all stores and departments, but due to limited understanding and time restrictions, it is out of scope for this research paper. Thirdly, originally the Kaggle competition asked to use Weighted Mean Average Error (WMAE) as a performance measure, it was not used in this research paper, as there are technically issue in implementing WMAE as a performance measure for both time series and regression in a grid search, besides, MAE, MSE, and RMSE seems to be sufficient.

References

Bonnes, K. (2014). Predictive analytics for supply chains: A systematic literature review. In *21st twente student conference on IT. Netherlands*.

Catal, C., Kaan, E. C. E., Arslan, B., & Akbulut, A. (2019). Benchmarking of Regression Algorithms and Time Series Analysis Techniques for Sales Forecasting. *Balkan Journal of Electrical and Computer Engineering*, *7*(1), 20-26.

Hülsmann, M., Borscheid, D., Friedrich, C. M., & Reith, D. (2012). General sales forecast models for automobile markets and their analysis. *Trans. MLDM*, *5*(2), 65-86.

Jain, A., Menon, M. N., & Chandra, S. (2015). Sales forecasting for retail chains.

Knott, B., Liu, H., & Simpson, A (2015). Predicting Sales for Rossmann Drug Stores.

Lasek, A., Cercone, N., & Saunders, J. (2016). Restaurant sales and customer demand forecasting: Literature survey and categorization of methods. In *Smart City 360°* (pp. 479-491). Springer, Cham.

Makatjane, K. D., & Moroke, N. D. (2016). Comparative study of holt-winters triples exponential smoothing and seasonal Arima: Forecasting short term seasonal car sales in South Africa. Risk Governance and Control: Financial Markets and Institutions, 6(1), 71-82.

Massaro, A., Maritati, V., & Galiano, A. (2018). Data Mining model performance of sales predictive algorithms based on RapidMiner workflows. *International Journal of Computer Science & Information Technology (IJCSIT)*, *10*(3), 39-56.

Mentzer, J. T., & Moon, M. A. (2004). *Sales forecasting management: a demand management approach*. Sage Publications.

Pavlyshenko, B. M. (2019). Machine-learning models for sales time series forecasting. *Data*, *4*(1), 15.

Prasetyo, E., Kusuma, A. K. Q. W., & Zainal, R. F. (2019). Shoes Sales Forecasting Using Autoregressive Integrated Moving Average (ARIMA) (Case Study UD. Wardana). Journal of Electrical Engineering and Computer Science*, Vol 3 Number 2, Dec 2018*, *3*(2).

Purthan, D., Shivaprasad, H., Kumar, K., & Manjunath, M. (2014). Comparing SARIMA and Holt-Winters' forecasting accuracy with respect to Indian motorcycle industry. Transactions on Engineering and Sciences, 2, 25-28.

Sumer, K. K., Goktas, O., & Hepsag, A. (2009). The application of seasonal latent variable in forecasting electricity demand as an alternative method. Energy Policy, 37(4), 1317-1322.

Udom, P. (2014). A comparison study between time series model and ARIMA model for sales forecasting of distributor in plastic industry. IOSR Journal of Engineering, 4(2), 32-38.

Wu, C. S. M., Patil, P., & Gunaseelan, S. (2018, November). Comparison of Different Machine Learning Algorithms for Multiple Regression on Black Friday Sales Data. In *2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)* (pp. 16-20). IEEE.